IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of

Gustavson, et al.

Serial No.: 10/671,888 Group Art Unit: 2193

Filed: September 29, 2003 Examiner: Ngo, C. D.

For: **METHOD AND STRUCTURE FOR PRODUCING HIGH PERFORMANCE LINEAR ALGEBRA ROUTINES USING REGISTER BLOCK DATA FORMAT ROUTINES**

Commissioner of Patents
Alexandria, VA 22313-1450

APPELLANTS' BRIEF ON APPEAL

Sir:

Appellants respectfully appeal the rejection of claims 1, 2, 4-9, 17-19, and 21-29 in the Office Action mailed on April 19, 2007. A Notice of Appeal was timely filed on July 18, 2007.

I. REAL PARTY IN INTEREST

The real party in interest is International Business Machines Corporation, assignee of 100% interest of the above-referenced patent application.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellants, Appellants' legal representative or Assignee which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1, 2, 4-9, 17-19, and 21-29, all of the claims presently pending in the application, stand rejected both as allegedly directed to nonstatutory subject matter and on prior art grounds. Claims 3, 10-16, and 20 are canceled.

IV. STATUS OF AMENDMENTS

An Amendment Under 37 CFR §1.116 was filed on June 19, 2007, presenting claim amendments to clarify the claimed invention in accordance with the conversation during the telephone interview dated June 13, 2007, and the rejection of record.

In the Advisory Action dated July 9, 2007, the Examiner indicated that the claim amendments raised a new issue and would not be entered, that Appellants' arguments were not persuasive, and that the rejections of record were maintained.

V. SUMMARY OF CLAIMED SUBJECT MATTER

As a summary, the present invention provides a mechanism to overcome a hardware deficiency in a computer, using a software mechanism wherein data is re-arranged preliminarily and stored in memory in a format that is optimum to overcome the deficiency and increase processing efficiency, particularly when the re-arranged data is repetitively retrieved from memory for the processing. In the exemplary embodiment described in the disclosure, the data is matrix data to be used in linear algebra subroutines, the deficiency of the hardware is the interface at the FPU, and the matrix data is desired to be finally loaded into the FPU in the format of the transpose of the original matrix data.

However, it is noted that the data stored in memory in the present invention is neither the original matrix data nor the transpose of the matrix data and is, therefore, not stored in standard format.

In general, the intermediate data format of the present invention is determined by considering what data format is optimal for the processing involved and the nature of the hardware deficiency and is arrived at by working backwards from the processing at the point of the hardware deficiency.

The bases in the specification for Appellants' invention is described in the following claim annotations.

1. (Rejected) A computer (Figs. 2-4), as controlled to implement a method of increasing efficiency in executing a matrix operation that uses matrix data in a standard format (See Fig 1 and label 601 of Fig. 6), said standard format comprising one of a column major format and a row major format, said method comprising:

for matrix data stored in said standard format (601, Fig. 6), wherein said matrix data comprises data of any of a complete matrix, a complete submatrix, or a part of a matrix or submatrix, separating said matrix data into blocks of data, each said block (603, 606 of Fig. 6) having a size p-by-q (lines 16-17 of page 19); and rearranging and placing in a storage of said computer (lines 17-20 of page 20), for retrieval for executing said matrix operation (lines 6-12 of page 20), said blocks of data to be contiguous blocks of contiguous data such that said matrix data is represented in a nonstandard format (see pseudomatrix blocks PA, item 605 of Figure 6 showing how the intermediate matrix of the present invention is different from either the original matrix data 601 and the transpose 602 that will ultimately be loaded into the FPU; PA blocks 605 are clearly stored in nonstandard format) that permits said matrix data to be moved from said storage into a position for performing said matrix operation more quickly than if said matrix data had been moved as stored in said standard format (601, Fig. 6).

2. (Rejected) The computer (Figs 2-4) of claim 22, wherein said co-processing unit comprises a floating point unit (FPU) and said loading said matrix data into said set of data registers comprises loading said blocks from said storage into a subset of data registers in said set of data registers, using a deviation from a normal floating point loading instruction of the floating point unit (FPU) of the computer (step 703, Fig. 7; lines 6-8 of page 9; lines 19-20 of page 14).

3. (Canceled)

4. (Rejected) The computer (Figs. 2-4) of claim 1, wherein said size p-by-q comprises a 2-by-2 block (605, Fig 6; line 14 of page 20).

5. (Rejected) The computer (Figs. 2-4) of claim 2, wherein said deviation from normal floating point loading comprises a crisscrossing of elements about a diagonal of said blocks (step 703, Fig 7; lines 6-9 of page 20; line 11 of page 18 through line 2 of page 19).

6. (Rejected) The computer (Figs. 2-4) of claim 2, said method further comprising:

selectively, at least one of loading input data and storing a result of said matrix operation into or out of said co-processing unit from L1 cache or memory by at least one of a subset of optimal load and store instructions, said loading and storing being dictated by an optimal FPU loading or storage instruction (lines 19-20 of page 14).

7. (Rejected) The computer (Figs 2-4) of claim 2, wherein said deviation of said normal floating point loading instruction, in combination with said nonstandard format, provides a result data of a transpose of said matrix data to reside in said data registers of said FPU (see Fig 6, comparing label 602 to 605; lines 5-8 of page 19).

8. (Rejected) The computer (Figs 2-4) of claim 2, wherein said loading comprises a 2 x 2 crisscrossing technique (step 703, Fig 7).

9. (Rejected) The computer (Figs 2-4) of claim 6, wherein said linear algebra operation comprises one of a BLAS kernel and a factorization kernel (Fig 1; lines 3-8 of page 11).

10-16. (Canceled)

17. (Rejected) A signal-bearing medium (Fig 8) tangibly embodying a program of machine-readable instructions executable by a digital processing apparatus to perform a method of storing information of a matrix in a register block data format, said method comprising:

receiving data for a matrix, said data comprising one of a complete matrix data, a complete submatrix data, and a partial matrix or submatrix data, said matrix data being stored in one of a standard column format and a standard row format;

dividing said matrix data into blocks, each said block having a size p-by-q (step 701 of Fig 7); and

at least one of:

storing elements in at least one of said blocks in at least one of a cache and a memory in a format in which is elements of said block occupy a location different from an original location in said block (605, Fig 6)

storing said blocks of size p-by-q in a memory in a format in which at least one said block occupies a position different from its original position in said matrix, said register data block format converting the matrix data to no longer be in either of said standard column format or said standard row format (605, Fig 6).

18. (Rejected) The signal-bearing medium (Fig 8) of claim 17, said method further comprising:

loading said blocks from said memory into a plurality of data registers so that a format of data in said data registers comprises a transpose data of said matrix (see Figure 6, comparing 605 with 602; lines 5-8 of page 19).

19. (Rejected) The signal-bearing medium (Fig 8) of claim 18, wherein said loading comprises a loading using a 2 x 2 crisscrossing technique (line 11 of page 18 through line 2 of page 19).

20. (Canceled)

21. (Rejected) The computer (Figs 2-4) of claim 1, wherein said matrix operation is executed on a co-processing unit (302, Fig 3) of said computer and said position for performing said matrix operation comprises a set of data registers 303 of said co-processing unit, said method further comprising:

retrieving said matrix data from said storage in said nonstandard format; and

loading said matrix data into at least a subset of said set of data registers in an optimal format, said optimal format comprising a format of said matrix data in said data registers such that a minimal possible time is required to utilize said matrix data in said data registers in said matrix operation in said co-processing unit (lines 6-8 of page 9; line 15 of page 18 through line 2 of page 19).

22. (Rejected) The computer (Figs 2-4) of claim 21, wherein said computer includes at least one of a machine architecture and an instruction set having one or more features that are less than optimal for executing said matrix operation, and said nonstandard format of matrix data and said optimal format in said data registers together provide a mechanism that overcomes said one or more features that are less than optimal for executing said matrix operation (line 11 of page 16 through line 21 of page 17).

23. (Rejected) A computer (Figs 2-4) configured to implement a method of increasing efficiency in executing a matrix operation that uses matrix data in a standard format, said standard format comprising one of a column major format and a row major format, said method comprising:

converting at least a part of said matrix data into a pseudo matrix format (605, Fig 6) comprising contiguous data that no longer represents said matrix data in said standard format, each pseudo matrix comprising a subset of said matrix data that is predetermined to permit a loading of said pseudo matrix data into a processing unit in an optimal format to perform said matrix operation, said optimal format comprising a format that allows a minimal possible time in said processing unit to utilize said matrix data in said matrix operation (lines 6-8 of page 9; lines 19-20 of page 14).

24. (Rejected) The computer (Figs 2-4) of claim 23, said method further comprising successively loading elements of each said pseudo matrix into said processing unit for executing said matrix operation, wherein said loading comprises successively placing data

of each said pseudo matrix into predetermined registers of a register set of said processor in said optimal format (lines 19-20 of page 19).

25. (Rejected) The computer (Figs 2-4) of claim 24, said method further comprising:

processing said matrix operation on said data in said optimal format, a result of said processing being stored in predetermined registers of said register set (line 15 of page 18 through line 8 of page 19); and

storing said result from said predetermined registers of said register set into memory in said pseudo matrix format (lines 9-10 of page 19).

26. (Rejected) A computer (Figs 2-4) having at least one of a machine architecture and an instruction set having one or more features that are less than optimal (lines 4-6 of page 3; lines 10-17 of page 4; line 14 of page 15 through line 8 of page 19) for executing a matrix operation, said computer configured to implement a method of overcoming said disadvantage, said method comprising:

rearranging at least a part of matrix data to be used in said matrix operation into a plurality of blocks (605 of Fig 6; lines 16-20 of page 4), each block having size p-by-q, such that said matrix data is no longer stored in a standard matrix format comprising one of a row major format and a column major format (note that blocks 603, 606 in pseudo matrix 605 no longer represent matrix A in standard format), said rearranged matrix data in said blocks being stored as contiguous blocks of contiguous data in a nonstandard format,

wherein said nonstandard format of said matrix data is predetermined to allow said matrix data to be placed into a processing unit for processing said matrix data in said matrix operation such that said disadvantage on said computer is overcome.

27. (Rejected) The computer (Figs 2-4) of claim 26, said method further comprising:

loading said matrix data in said nonstandard format into at least a subset of data registers of said processing unit in an optimal format, said optimal format comprising a format allowing a minimal possible time in said processing unit to utilize said matrix data

in said matrix operation (step 703, Fig 7; line 20 of page 7 through line 3 page 8; lines 18-20 of page 14).

28. (Rejected) A computer (Figs 2-4) configured to implement a method of overcoming a hardware disadvantage on said computer relative to a specific processing on a specific computer architecture/set of instructions, said method comprising:

using first software instructions to preliminarily process input data to be used in said specific processing on said specific computer architecture/set of instructions in a manner to generate a first error relative to said specific processing (steps 701, 702, Fig 7); and

using second software instructions to subsequently process said input data in a manner to generate a correcting error relative to said specific processing (step 703, Fig 7),

wherein first software instructions in combination with said second software instructions overcome said disadvantage (lines 1-3 of page 8; lines 6-8 of page 9; lines 19-20 of page 14; lines 6-7 of page 23).

29. (Rejected) The computer (Figs 2-4) of claim 30, wherein said specific processing comprises a matrix operation and said first error comprises storing matrix data in a format that converts matrix data from a standard column major or row major format into a nonstandard format predetermined to overcome said disadvantage when said data is subjected to said correcting error (process 700 of Fig 7; lines 6-8 of page 9; lines 19-20 of page 14).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Appellant presents the following issues for review by the Board of Patent Appeals and Interferences:

ISSUE 1: The Statutory Subject Matter Rejection for Claims 1, 2, 4-9, 17-19, and 21-29

ISSUE 2: The Indefiniteness Rejection for Claims 1, 2, 4-9, and 21-29

ISSUE 3: The Anticipation Rejection for Claims 1, 4, 17-19, and 21-29, Based on US Patent 7,031,994 to Lao et al.

ISSUE 4: The Obviousness Rejection for Claims 2 and 5-9, based on Lao et al.

VII. ARGUMENTS

ISSUE #1: THE STATUTORY SUBJECT MATTER REJECTION

The Examiner alleges that all claims are directed to non-statutory subject matter.

In the Office Action mailed April 19, 2007, the Examiner characterizes that claims "*... 1, 2, 4-9 and 21-29 are directed to a computer implementing data manipulation method. Claims 17-19 are directed to a computer readable medium having instruction[s] implementing the method.*"

The Examiner continues: "*In order for such a computer related invention to be statutory, the claimed invention must accomplish a practical application. That is the claimed invention must transform an article or physical object to a different state or thing, or produce a useful, concrete and tangible result.... It is clear from the claims that the invention merely involves data manipulations for rearranging data of a matrix. The claimed invention does not an (sic) transform an article or physical object to a different state or thing, and the result produced by the claimed invention is a mere set of data being arranged from an original data matrix and does not have a real world value, and thus is not a (sic) useful, concrete and tangible. Therefore, the claimed invention is directed to non-statutory subject matter for failing to accomplish a practical application. Claims 17-19 are also rejected under 35 U.S.C. 101 as being directed to [a] signal carrier which is non-statutory subject matter.*"

Beginning in reverse order, claims 17-19 are, by the clear language of the claims themselves, directed to "a signal bearing medium tangibly embodying a program of machine-readable instructions executed by a digital processing apparatus to perform a method of storing information of a matrix in a register block format ...." Contrary to the Examiner's allegation, the plain meaning of the claim language clearly describes that instructions are tangibly embodied on a signal bearing medium. As such, these claims are not directed to a signal carrier, as alleged in the rejection.

More significant, Appellant submits that these claims are directed to Beauregard-type claims, conceded by the USPTO to be statutory subject matter in *In Re Beauregard*, 53 F.3d 1583 (1995), as demonstrated by subsequently-issued U.S. Patent No. 5,710,578 to Beauregard et al., dated January 20, 1998.

Relative to the remainder of the claims, claims 1, 2, 4-9, and 21-30 are clearly directed to a computer configured to implement the method of the invention, and, therefore, not subject to the analysis for computerized-method claim analysis. However, if these claims were to be subject to such analysis, Appellants submit that the present invention clearly passes evaluation under either prong of the test presented in the rejection.

That is, according to the first prong, the present invention clearly uses a software mechanism to transform the state of the computer from an inefficient processing state due to a deficiency in hardware (e.g., an inefficient interface at the L1/FPU) into a state predetermined to efficiently execute the desired processing, by using data in a reorganized order. This change in data in memory constitutes a physical change to the machine, since the machine state change is physically measurable. Accordingly, because of this state transformation of a piece of computing machinery, the present invention, therefore, clearly transforms an article and passes the first prong.

According to the second prong, in considering the present invention as a computerized method, Appellants submit that the result of the present invention is inherently real-world, practical, useful, concrete, and tangible. That is, the result of the present invention is to provide a software mechanism to overcome a hardware deficiency at the L1/FPU interface, as well as improve efficiency in the processing. Conventional wisdom would have addressed this problem by a redesign of the hardware.

The inventors recognized, however, that this problem might be corrected using software rather than an expensive hardware redesign. In implementing the method of the invention for the specific problem on the assignee's BlueGeneL computer as implemented to process linear algebra computations and using newer designs of FPUs, they ultimately realized that a more generalized concept was involved wherein two "errors" are sequentially executed that together provide a solution at least as efficient as if the hardware deficiency were not present.

Therefore, the present invention inherently provides the real world result of correcting a hardware deficiency, as well as improving efficiency in processing with the FPU current design. Either one of these two results demonstrate a real-world, practical application, thereby clearly passing the second prong.

Appellants respectfully submit that an underlying logical flaw in the evaluation of record is that the Examiner considers that statutory subject matter is established if an Examiner is able to devise any characterization of an invention that allegedly places it into the category of non-statutory subject matter. Therefore, using this rationale, according to the Examiner, the present invention is non-statutory because "*[i]t is clear from the claims that the invention merely involves data manipulations for rearranging data of a matrix.*"

However, the Examiner cites no case law supporting this articulation of the test for statutory subject matter, and Appellants submit that the rejection of record fails to meet the initial burden of establishing a lack of statutory subject matter.

That is, whether an Examiner chooses to arbitrarily characterize the present invention as "*... merely ... rearranging data of a matrix*" is irrelevant, if the actual result (e.g., the data rearrangement) improves processing efficiency on a machine and/or overcomes the design deficiency of a hardware interface, since the result inherently satisfies the standard test for statutory subject matter.

Moreover, it is noted that "useful, concrete and tangible result" test for statutory subject matter for computerized methods arose out of the concern for precluding mathematical algorithms from being claimed as abstract ideas. The present invention does not involve an attempt to claim a mathematical algorithm, either in the abstract or in a practical application.

Therefore, relative to the Examiner's characterization that the reorganized matrix data "*... does not have a real world value ...*", Appellants submit that the improvement in processing efficiency and the overcoming of a hardware design deficiency inherently provides a real world value. Other real world applications are discussed beginning at line 21 on page 26 of the specification.

It is noted that Appellants have tried several times to work with the Examiner to find specific wording that would be more palatable for this specific Examiner with no

success, presumably because of the Examiner's perception that statutory subject matter is determined by the Examiner's arbitrary characterization of an invention, rather than the subject matter itself as demonstrated to inherently have a real world application.

Therefore, Appellants respectfully request that the Board reverse the statutory subject matter rejection of record and, if necessary, remand back to the Examiner for purpose of working out any wording changes deemed by the Board to be necessary. However, along this line, Appellants believe that it is the subject matter itself that is important in the determination of statutory subject matter, not any specific set of wording in the claims.

ISSUE #2: THE INDEFINITENESS REJECTION FOR CLAIMS 1, 2, 4-9, and 21-29

1.) The Examiner considers that claims 1, 23, 26, and 28 are rendered indefinite because "*... the claim appears directed to an apparatus, but fails to recite a combination of physical means to define the apparatus. Further, since the body of the claims recite[s] only steps of a method, it is unclear what category the invention belongs. Claims 23, 26, and 28 have the same problem.*"

In response, Appellants submit that the claims were revised in an attempt to focus the evaluation on an apparatus rather than method claims and that a subsequent attempt to add claim language adding component descriptions further defining the computer were rejected by the Examiner as raising new issues.

Therefore, Appellants submit that, should the statutory subject matter and prior art rejections be resolved in Appellants' favor, this rejection is easily resolved by wording changes, if necessary.

However, Appellants respectfully submit that these claims, even in their present state, are perfectly acceptable, since they clearly address an apparatus that is defined in terms of a configuration defined functionally, and that such functional definition is completely proper for claim language.

2.) The Examiner considers that claim 2 is indefinite because of the wording "using a deviation from a normal floating point loading instruction."

In response, Appellants respectfully submit that this terminology is clearly explained in several places in the specification, including at least lines 5-6 of page 3, line 20 of page 7 through line 2 of page 8, and perhaps most significantly, at lines 18-20 of page 16, giving rise to the "crisscross" description in the claim language, and lines 12-15 of page 17.

Therefore, Appellants submit that the disclosure provides adequate support for this claim terminology and respectfully request that the Board remove this rejection.

3.) The Examiner considers that claim 28 is indefinite because of the description "... a method of overcoming a hardware disadvantage on said computer relative to a specific processing on a specific computer architecture/set of instructions", along with "a first error relative to said specific processing" and "a correcting error relative to said processing."

In response, Appellants submit that this terminology is well described in the disclosure, including at least at lines 1-3 of page 8, line 17 of page 15 through line 20 of page 16, lines 7-15 of page 17, lines 3-8 of page 19, and line 4 of page 24 through line 7 of page 25.

Accordingly, Appellants submit that the disclosure provides adequate support for this terminology and respectfully request that the Board reverse this rejection.

4.) The Examiner correctly notes that claim 29 has a typographical error in referring back to claim 30 and, although the Examiner declined to enter an amendment to correct such error, Appellants believe that correction of this error will be easy upon remand.

ISSUE #3: THE ANTICIPATION REJECTION FOR CLAIMS 1, 4, 17-19 AND 21-29 BASED ON LAO ET AL (US PATENT 7,031,994)

The Examiner considers that Lao et al anticipates the present invention described by claims 1, 4, 17-19, and 21-29, presumably because the exemplary embodiment in the present application demonstrates the method of the present invention as implemented to load the transpose of the matrix into the FPU.

However, Appellants respectfully submit that the method of the claimed invention is clearly patentably distinguishable from the transposition process described in Lao, particularly in view of the different problem being addressed by the present invention from that of Lao. More specifically, Lao addresses simply the process of matrix transposition as a solution to save memory space requirement (see lines 46-47 of column 1 of Lao).

To achieve this transposition, as clearly described even in the Abstract, Lao partitions the matrix data into blocks, moves these blocks row-wise into the cache, rewrites the data row-wise into a column of memory and uses a permutation vector to arrive at the transpose of the original matrix data.

Appellants submit that this simple matrix transposition fails to satisfy the plain meaning of the claim language of even the independent claims, particularly in view of the different purpose of the present invention, as articulated in the claims themselves.

That is, in contrast to the simple transposition method of Lao, the exemplary embodiment described in the disclosure arrives at the transpose of portions of the matrix data only after it has been loaded into the FPU, using the non-standard crisscross loading pattern (e.g., reference dependent claim 18). The independent claims of the present application actually refer to the "pseudo matrix" PA 605 shown in Figure 6 that is stored in memory to be repetitively retrieved for the matrix processing being executed in the FPU. Upon retrievable and loading into the FPU using the crisscross loading pattern, this retrieved pseudo matrix data appears in the L0 registers of the FPU in the format of transposed matrix portions useful for the matrix processing.

Therefore, the rearrangement of data in memory in the present invention does not result in the transpose of the matrix being stored in memory, as occurs in Lao. Rather, the

pseudo matrix is stored in memory, to be retrieved repetitively for use by the FPU processing.

Lao makes no suggestion for storing matrix data in any format except the matrix transpose and, indeed, such matrix transposition is the entire purpose of Lao. Thus, Lao does not store anything in memory except either the original matrix data or the matrix transposition data. Both of these formats are in standard format.

In contrast, the present invention stores a pseudo matrix in memory (for repetitive retrieval) and this data is not stored in standard format (see Figure 6). Because of this fundamental difference in purpose and the failure to store a pseudo matrix in nonstandard format, the technique of Lao fails to satisfy the plain meaning of the claim language, as follows.

Relative to independent claim 1, Lao stores the matrix transpose only. There is nothing stored in Lao that is in nonstandard format. In the environment of the L1 cache/FPU interface problem of the present invention, this simple transposed matrix data would have the same inefficiency noted by the present inventors, which was addressed using the software method of using two "errors" to place the matrix data into the FPU registers in the desired format of the transpose. Therefore, assuming *arguendo* that the transposed matrix data of Lao were to be subsequently retrieved from memory for processing (as required by the plain meaning of the claim language), there is no suggestion in Lao to store blocks of data in nonstandard format (since transposed matrix data is still standard format). Nor is there any suggestion in Lao of an architecture for which matrix data stored in nonstandard format would be beneficial, as required by the final claim limitation.

More specifically, the matrix processing of the present invention is being executed in the FPU. Even if the transposition processing in Lao were to be considered the "matrix operation" required by the language of claim 1, there is nothing in Lao corresponding to storing the matrix data in a memory in nonstandard format, since the processing of moving rows of data in Lao is occurring via the cache and the movement of these blocks of row data would not qualify to satisfy the plain meaning of the claim language requiring that data be stored in nonstandard format. The Examiner points to the movement of data

demonstrated in column 16 of Lao that is used to explain the formation of a matrix transpose. However, this data movement is not equivalent to storing the data in nonstandard format, and the only significant storage of data in Lao is the storage of the transposed matrix, which is stored in standard format.

In summary, Lao is not intending to store matrix data in nonstandard format for purpose of performing a matrix operation, as required by the plain meaning of independent claim 1. In contrast, the present invention actually intends to retrieve this pseudo matrix data repetitively as part of the linear algebra processing in the FPU, thereby adding to the efficiency of the method of the present invention.

Independent claim 17 has similar distinction from Lao in its wording.

Relative to independent claims 23 and 26 and dependent claims 24, 25, and 27, there is no equivalent in Lao of a pseudo matrix data that is stored in nonstandard format and that is loaded into a processing unit, let alone a pseudo matrix predetermined as permitting an optimal loading and an optimal processing time for linear algebra processing. Lao also fails to describe a disadvantage with hardware or instructions, as required by claim 26, or of placing data into registers, as required by claims 24, 25, and 27.

Relative to independent claim 28 and dependent claim 29, there are no errors being executed in Lao. Nor is there any hardware disadvantage being overcome relative to a matrix processing.

Relative to dependent claims 4, 8, , the 2x2 block in Lao is used only for generating the transpose, which is stored in standard format.

Relative to dependent claim 9, Lao does not discuss using the transpose in a linear algebra operation, since the discussion at lines 39-47 of column 1 does not get into details related to the applications. It is clear, however, that the technique of forming the transpose upon loading into the FPU is not suggested in Lao, since the transpose is already stored in memory and ready for retrieval for the applications discussed in column 1.

Relative to dependent claims 18 and 19, the matrix transpose in Lao is loaded into memory itself, not data registers, let alone using a crisscross loading pattern. Lao does not address the problem of loading into FPU data registers.

Relative to dependent claim 21, Lao does not discuss matrix operation on a co-processor, retrieval of matrix data stored in nonstandard format or loading the data stored in nonstandard format into a co-processor.

Relative to dependent claim 22, Lao is not concerned with overcoming a problem with the matrix processing.

Because Lao fails to demonstrate each of the above-identified elements of the claims, the rejection currently of record fails to meet the initial burden of a *prima facie* rejection.

ISSUE #4: THE OBVIOUSNESS REJECTION FOR CLAIMS 2 AND 5-9, ALSO BASED ON LAO

The Examiner considers that Lao renders obvious claims 2 and 5-9 even though there is no other processing described in Lao except the matrix transposition via cache loading/unloading.

Appellants respectfully submit that the Examiner has failed to meet the initial burden of an obviousness rejection for these claims, since the entire purpose of Lao is to form the matrix transpose, using cache loading/unloading. There is no suggestion in Lao of the details of these claims.

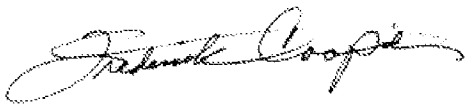
That is, relative to dependent claims 2, 5, 6,7, Lao does not perform any processing in an FPU, let alone a suggestion to use a deviation from a normal loading instruction. Lao simply achieves the matrix transpose in a manner drastically different from that of the claimed invention. This different procedure is expected since Lao is not concerned with overcoming a problem of a hardware interface on its machine.

CONCLUSION

In view of the foregoing, Appellants submit that claims 1, 2, 4-9, 17-19, and 21-29, all the claims presently pending in the application, are clearly enabled and patentably distinct from the prior art of record and in condition for allowance. Thus, the Board is respectfully requested to remove all rejections of claims 1, 2, 4-9, 17-19, and 21-29.

Please charge any deficiencies and/or credit any overpayments necessary to enter this paper to Assignee's Deposit Account number 50-0510.

Respectfully submitted,



Dated: September 18, 2007

Frederick E. Cooperrider
Reg. No. 36,769

McGinn Intellectual Property Law Group, PLLC.
8231 Old Courthouse Road, Suite 200
Vienna, VA 22182-3817
(703) 761-4100
Customer Number: 21254

VIII. CLAIMS APPENDIX

Claims, as reflected upon entry of the Amendment Under 37 CFR §1.111 filed on January 25, 2007:

1. (Rejected) A computer, as controlled to implement a method of increasing efficiency in executing a matrix operation that uses matrix data in a standard format, said standard format comprising one of a column major format and a row major format, said method comprising:

for matrix data stored in said standard format, wherein said matrix data comprises data of any of a complete matrix, a complete submatrix, or a part of a matrix or submatrix, separating said matrix data into blocks of data, each said block having a size p-by-q; and rearranging and placing in a storage of said computer, for retrieval for executing said matrix operation, said blocks of data to be contiguous blocks of contiguous data such that said matrix data is represented in a nonstandard format that permits said matrix data to be moved from said storage into a position for performing said matrix operation more quickly than if said matrix data had been moved as stored in said standard format.

2. (Rejected) The computer of claim 22, wherein said co-processing unit comprises a floating point unit (FPU) and said loading said matrix data into said set of data registers comprises loading said blocks from said storage into a subset of data registers in said set of data registers, using a deviation from a normal floating point loading instruction of the floating point unit (FPU) of the computer.

3. (Canceled)

4. (Rejected) The computer of claim 1, wherein said size p-by-q comprises a 2-by-2 block.

5. (Rejected) The computer of claim 2, wherein said deviation from normal floating point loading comprises a crisscrossing of elements about a diagonal of said blocks.

6. (Rejected) The computer of claim 2, said method further comprising:

selectively, at least one of loading input data and storing a result of said matrix operation into or out of said co-processing unit from L1 cache or memory by at least one of a subset of optimal load and store instructions, said loading and storing being dictated by an optimal FPU loading or storage instruction.

7. (Rejected) The computer of claim 2, wherein said deviation of said normal floating point loading instruction, in combination with said nonstandard format, provides a result data of a transpose of said matrix data to reside in said data registers of said FPU.

8. (Rejected) The computer of claim 2, wherein said loading comprises a 2 x 2 crisscrossing technique.

9. (Rejected) The computer of claim 6, wherein said linear algebra operation comprises one of a BLAS kernel and a factorization kernel.

10-16. (Canceled)

17. (Rejected) A signal-bearing medium tangibly embodying a program of machine-readable instructions executable by a digital processing apparatus to perform a method of storing information of a matrix in a register block data format, said method comprising:

receiving data for a matrix, said data comprising one of a complete matrix data, a complete submatrix data, and a partial matrix or submatrix data, said matrix data being stored in one of a standard column format and a standard row format;

dividing said matrix ~~A~~ data into blocks, each said block having a size p-by-q; and

at least one of:

storing elements in at least one of said blocks in at least one of a cache and a memory in a format in which is elements of said block occupy a location different from an original location in said block

storing said blocks of size p-by-q in a memory in a format in which at least one said block occupies a position different from its original position in said matrix, said register data block format converting the matrix data to no longer be in either of said standard column format or said standard row format.

18. (Rejected) The signal-bearing medium of claim 17, said method further comprising:

loading said blocks from said memory into a plurality of data registers so that a format of data in said data registers comprises a transpose data of said matrix.

19. (Rejected) The signal-bearing medium of claim 18, wherein said loading comprises a loading using a 2 x 2 crisscrossing technique.

20. (Canceled)

21. (Rejected) The computer of claim 1, wherein said matrix operation is executed on a co-processing unit of said computer and said position for performing said matrix operation comprises a set of data registers of said co-processing unit, said method further comprising:

retrieving said matrix data from said storage in said nonstandard format; and

loading said matrix data into at least a subset of said set of data registers in an optimal format, said optimal format comprising a format of said matrix data in said data registers such that a minimal possible time is required to utilize said matrix data in said data registers in said matrix operation in said co-processing unit.

22. (Rejected) The computer of claim 21, wherein said computer includes at least one of a machine architecture and an instruction set having one or more features that are less than optimal for executing said matrix operation, and said nonstandard format of matrix data and said optimal format in said data registers together provide a mechanism that overcomes said one or more features that are less than optimal for executing said matrix operation.

23. (Rejected) A computer configured to implement a method of increasing efficiency in executing a matrix operation that uses matrix data in a standard format, said standard format comprising one of a column major format and a row major format, said method comprising:

converting at least a part of said matrix data into a pseudo matrix format comprising contiguous data that no longer represents said matrix data in said standard format, each pseudo matrix comprising a subset of said matrix data that is predetermined to permit a loading of said pseudo matrix data into a processing unit in an optimal format to perform said matrix operation, said optimal format comprising a format that allows a minimal possible time in said processing unit to utilize said matrix data in said matrix operation.

24. (Rejected) The computer of claim 23, said method further comprising successively loading elements of each said pseudo matrix into said processing unit for executing said matrix operation, wherein said loading comprises successively placing data of each said pseudo matrix into predetermined registers of a register set of said processor in said optimal format.

25. (Rejected) The computer of claim 24, said method further comprising:

processing said matrix operation on said data in said optimal format, a result of said processing being stored in predetermined registers of said register set; and

storing said result from said predetermined registers of said register set into memory in said pseudo matrix format.

26. (Rejected) A computer having at least one of a machine architecture and an instruction set having one or more features that are less than optimal for executing a matrix operation, said computer configured to implement a method of overcoming said disadvantage, said method comprising:

rearranging at least a part of matrix data to be used in said matrix operation into a plurality of blocks, each block having size p-by-q, such that said matrix data is no longer stored in a standard matrix format comprising one of a row major format and a column major format, said rearranged matrix data in said blocks being stored as contiguous blocks of contiguous data in a nonstandard format,

wherein said nonstandard format of said matrix data is predetermined to allow said matrix data to be placed into a processing unit for processing said matrix data in said matrix operation such that said disadvantage on said computer is overcome.

27. (Rejected) The computer of claim 26, said method further comprising:

loading said matrix data in said nonstandard format into at least a subset of data registers of said processing unit in an optimal format, said optimal format comprising a format allowing a minimal possible time in said processing unit to utilize said matrix data in said matrix operation.

28. (Rejected) A computer configured to implement a method of overcoming a hardware disadvantage on said computer relative to a specific processing on a specific computer architecture/set of instructions, said method comprising:

using first software instructions to preliminarily process input data to be used in said specific processing on said specific computer architecture/set of instructions in a manner to generate a first error relative to said specific processing; and

using second software instructions to subsequently process said input data in a manner to generate a correcting error relative to said specific processing,

wherein first software instructions in combination with said second software instructions overcome said disadvantage.

29. (Rejected) The computer of claim 30, wherein said specific processing comprises a matrix operation and said first error comprises storing matrix data in a format that converts matrix data from a standard column major or row major format into a nonstandard format predetermined to overcome said disadvantage when said data is subjected to said correcting error.

IX. EVIDENCE APPENDIX

(NONE)

X. RELATED PROCEEDINGS APPENDIX

(NONE)